

DIDACTIC SIMULATORS FOR UNDERSTANDING ROUTING PROTOCOLS IN WIRELESS SENSOR NETWORKS

A. Paradisi¹, M.P. Uwase², J. Tiberghien³, K. Steenhaut⁴, J.M. Dricot⁵

¹ *Vrije Universiteit Brussel (GREECE)*

² *Vrije Universiteit Brussel, Université Libre de Bruxelles (RWANDA)*

³ *ETRO - Vrije Universiteit Brussel (BELGIUM)*

⁴ *IWT - Erasmus Hogeschool Brussel, ETRO Vrije Universiteit Brussel (BELGIUM)*

⁵ *OPERA - Université Libre de Bruxelles (BELGIUM)*

artparad@gmail.com, marie.paule.uwase@vub.ac.be, jacques.tiberghien@vub.ac.be,
ksteenha@etro.vub.ac.be, jean-michel.dricot@ulb.ac.be

Abstract

Wireless sensor networks consist of autonomous intelligent sensor nodes, usually powered by battery, that can measure certain characteristics of their environment, such as temperature, pressure, moisture, acceleration, etc. These sensor nodes communicate by radio with their neighbours to forward data to a central collection point from where it is sent to some analysis centre via the Internet. The use of batteries and radio communications minimizes the cost, but requires specific routing protocols. Wireless Sensor Networks fit in the broader category of “smart object networks” that are gaining attention as the large range of IPv6 addresses will make possible “the Internet of Things”. In this perspective the Internet Engineering Task Force (IETF) has created a working group in charge of the design of a new routing protocol for low power and lossy networks called RPL. The authors are regularly faced with masters students interested in wireless sensor networks as a research topic for their thesis, but not knowing enough about it to make a rational choice. For that reason, a very simple, user friendly, interactive simulator that exposes the underlying principles of the Distance Vector (DV), the Ad-hoc, On Demand Distance Vector (AODV) and the RPL protocols has been developed and is being evaluated with master's students. It does not replace in any sense the more sophisticated simulation environments that are available to researchers. On the contrary, it is used to clarify basics of routing before confronting the students with the fully fledged tools. This simulator is now used as a support for lectures on routing and as a self-teaching tool for students considering a master's thesis in two universities, one in Europe and one in Africa. Its didactic value is being quantitatively evaluated with the participants of an international summer school for engineering students. In this paper, the decision to build a didactic tool rather than adapting existing tools is motivated and the design issues are discussed, with emphasis on RPL. The results of the evaluation of the didactic qualities by various groups of students in different educational contexts will also be presented.

Keywords: Didactic simulations, wireless sensor networks, routing, RPL.

1 INTRODUCTION

In this introduction wireless sensor networks and the Internet of Things (IoT) will be briefly introduced and the specific requirements for routing in these networks explained.

1.1 Wireless Sensor Networks (WSN)

Wireless sensor networks are made with autonomous intelligent sensors, usually powered by battery, which can measure certain characteristics of their environment, such as temperature, pressure, noise level, acceleration, etc. These sensors (often called “motes”) communicate by radio with their neighbours to forward their measurements to a central data collection point from where all the data is sent to some processing and analysis center via any classical network, such as the Internet. The use of batteries and radio communications allow to minimize the deployment cost of this distributed measurement system. Wireless sensor networks have a broad range of applications such as climate monitoring, tropical

flood prevention, early detection of bush fires, wildlife herds monitoring, detection of alarming behaviours of elderly people living alone, power monitoring in computer centres. A clear overview of different varieties of sensor networks and their applications was published by Kay Römer and Friedemann Mattern in 2004 [1]. Fairly extensive and up to date overviews of existing wireless sensor nodes are available from The Sensor Network Museum [2] and from the Mini Hardware Survey by Tatania Bokareva [3]. A typical sensor node contains a 16 bit ultra-low power microprocessor (such as the TI MSP430 or the Intel ARM), running at a clock frequency in the 10 MHz range, some Kilobytes of static RAM and 48 to 128 Kbytes of Flash program memory. The processor controls a radio transceiver and appropriate input-output devices for sensing the phenomena the network should monitor. Temperature, humidity, pressure and acceleration sensors are common, one or more analog-to-digital (AD) converters allow addition of other sensors. Ports to interface a GPS receiver are sometimes provided.

1.2 The Internet of Things

The last few years the scope of sensor networks is growing considerably: the cost of simple nodes has decreased considerably, much more powerful hardware becomes affordable, and the environmental concerns (especially in the field of energy conservation and management) call for a broad range of new applications [4]. The initial sensor networks were stand-alone networks or networks connected to the Internet through an application gateway (see fig.1). They used highly specific communication protocols, often poorly structured, that made mixing of components of different brands difficult.

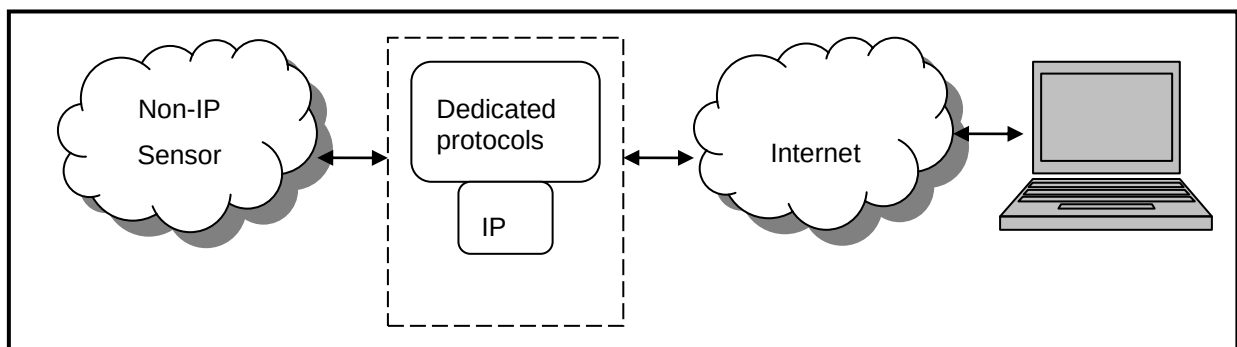


Figure 1. Connecting a WSN to the Internet via an application gateway.

Some order was brought in through the IEEE 802.15.4 standard that promotes an ad-hoc architecture and protocols for the physical, medium access control and link layers [5]. After this first step towards standardisation, the advent of IPv6, with its almost infinite range of addresses, was a good opportunity to further standardise WSNs and to facilitate their integration in the Internet, without the need for cumbersome application gateways (see fig.2). For that purpose, the Internet Engineering Task Force (IETF) chartered in 2005 the 6LoWPAN working group to standardise the use of IPv6 over IEEE 802.15.4 systems. This working group has focused on two topics: (i) how to carry IPv6 datagrams in IEEE 802.15.4 frames and (ii) how to perform the IPv6 neighbour discovery functions in a network with overlapping broadcast domains. Routing was outside the scope of this working group. In 2008, another working group, called “Routing over Low-power and Lossy network” (ROLL) was created. This working group studied extensively the existing routing protocols in use in the Internet and those specifically designed for WSNs. It concluded that none of the existing protocols completely fulfilled all the requirements for the operation of a real-life WSN. Therefore the working group proposed a new protocol, i.e., the “Routing Protocol for Low-power and lossy network” (RPL) [6].

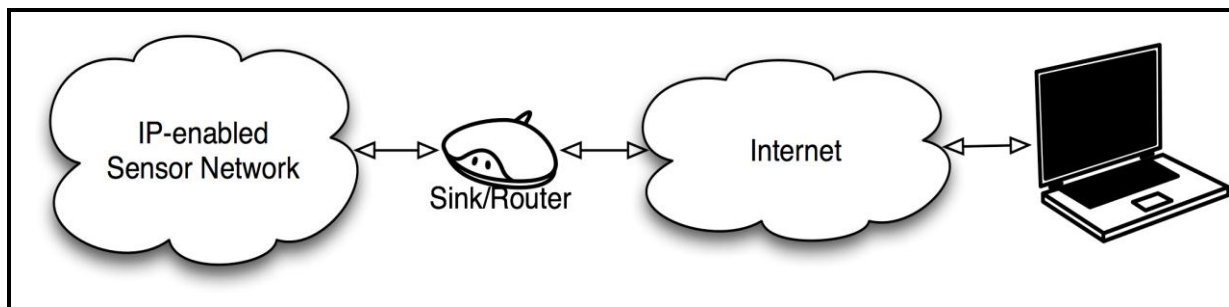


Figure 2. Integration of sensor networks and the Internet through the use of IPv6.

This paper presents the development of a new didactic tool intended for helping students understanding the underlying mechanisms of RPL.

1.3 Routing in Low power and Lossy Networks.

A routing protocol is a set of algorithms or procedures whose prime role is to establish a path between source and sink, to keep track of the availability of such route and to facilitate successful transmission of data along the chosen route. The routes should be selected so that data reaches its destination in the 'best' way possible. The optimality is defined by one or more metrics, depending on the application requirements. A widely used metric is using the route with the lowest delay, or the highest throughput. Several other approaches are based on the use of the route with the least hop distance, the best link quality, or the least energy consumption, among others. Low power and Lossy Networks (LLNs) are a class of networks where nodes are constrained: LLN routers typically operate with constraints on processing power, memory and energy. In wireless communications, packet drops on lossy links are extremely frequent, and the links may become completely unusable for a long period of time for a number of reasons such as the presence of interference, slow fading, or line-of-sight obstruction. This observation has strong consequences on the design of routing protocols [6]. As typical LLNs have many nodes and only one or a few gateways to the outside world (often the Internet), one can distinguish two main classes of routing protocols: (i) the "dissemination" protocols, intended for sending software or commands from a gateway to all, or some, sensors and (ii) the "collect" protocols, used by external processors for acquiring, via the gateways, the data generated in the sensors [7]. In almost every LLN applications, the collect protocol carries the largest part of the data traffic, and its efficiency is, therefore, a primary concern. The dissemination protocol is typically used for bootstrapping the LLN and for setting parameters in specific nodes. Therefore its reliability is more important than its efficiency. Some networks allow sensor-to-sensor traffic. This can be implemented by using the collect protocol immediately followed by the disseminate protocol. It avoids a lot of routing complexity at the cost of additional traffic to and from the gateway nodes. As sensor-to-sensor traffic seems to be rather exceptional this added traffic is not an issue [4]. By restricting routing to the collect and disseminate protocols one considerably reduces the amount of routing data to be stored in each node. For the collect protocol, only routing information to reach the gateways needs to be stored and for the disseminate protocol, provided it is seldom used, a stateless hop-by-hop broadcasting protocol could be sufficient.

2 SIMULATORS AS DIDACTIC TOOLS.

To be up-to-date with recent developments in engineering and technology, and to develop a good understanding of the underlying principles, students are now required to play an active role in the learning process. Participating in interactive learning activities, such as simulations, makes students pay more attention and guarantees better understanding of the theoretical concepts. Many simulators providing a very realistic experience within the field of LLNs are available. More specifically, some of the well-known simulation tools used by researchers are: Cooja [8], made available by the Contiki developers to facilitate the

development and testing of sensor networks and the OMNET++ [9] an open source simulator for networked computer systems in general. These simulators are useful tools for researchers that are already familiar with the field of wireless communications, but not for students without any prior knowledge on LLNs. Although it seems reasonable that understanding languages and tools like these is valuable for many students, it is hard to motivate them to learn the simulation software before they know what they can use it for. This is why it was decided to develop a low threshold didactic simulator for the self-learning of basic routing. This was done by two students as partial fulfilment of the requirements for their master's thesis in applied computer science. The requirements and didactic trade-offs are described in the following sections.

2.1 Scope

The first issue that needs to be dealt with when designing didactic tools is how to make straightforward real world problems without becoming trivial and, more important, without giving the user an incorrect image of the real problem. A first simplification consists in limiting the scope of the simulation by restricting the number of gateways to one and by considering only the collect protocol. This considerably reduces the complexity of routing and the graphical user interface, without affecting the underlying algorithms. Also, it puts forward the most critical issues without distracting the student with complex algorithms that have little influence on the overall performance of the LLN. In a later version, dissemination protocols might be added.

2.2 Protocols

Even if the final goal of the simulator is to familiarize students with the algorithms underlying the RPL protocol, other, less elaborate, protocols should also be available to introduce the basics of routing. As most collect protocols in LLNs are sophisticated extensions of the Distance Vector (DV) protocol [10,11], it seems necessary to propose a bare version of this protocol to students discovering routing issues. The Ad-hoc On-demand Distance Vector (AODV) protocol [12] was also included as a simple example of reactive protocol, i.e., it establishes routes only when needed while the classical distance vector protocol updates its routing information periodically, regardless of the actual need. The simulation of the RPL protocol [13] was somewhat more complex, as for repairing the collect routing tables in case of node or link failure it uses mechanisms belonging to the disseminate part of the protocol. In fact both the collect and disseminate protocols needed to be implemented in the simulator, but, in the present version of the simulator, only the collect function is shown to the user.

2.3 Route quality metric

As mentioned earlier, routing algorithms try to find the optimal route to the gateway, the criteria for optimality depending on the specific application. For this didactic simulator, the user can assign to each link an integer "cost" figure and the best route is the one with the lowest cost. This cost can represent for instance a physical distance (but is more likely to be a quantification of the quality of the radio link such as the average number of retries necessary to transmit a message over that link). In RPL, an additional concept is introduced to prevent loops in the routing algorithm: it is called the rank of a node. When a route is followed from a given node to the gateway, the rank of the encountered nodes should always decrease along the route. This metric is loosely coupled with the cost of a route, but the relationship is not specified in the proposed standard. It is stated that the rank should be linked to the cost by an application specific objective function. For the didactic simulator, the number of hops to the gateway has been chosen as rank.

2.4 Link and node failures

Variations in link quality and premature failure of some nodes are an inherent property of LLNs. The user can modify at any time during a simulation the cost attribute of a link. Giving

it a very high cost figure can simulate unavailability of. For simulating the failure of nodes, an integer-valued “battery” attribute has been assigned to each node. Most operations involving a node cause the value of that battery attribute to decrease. When the value of a battery attribute reaches a predefined threshold, the node and all its links are disabled. During a simulation run, the user can modify the battery state of any node. Restricting failure causes to battery exhaustion is a simplification that does not restrict the study of the effects of node failure in general.

2.5 Timers

Timers play an important role in many routing algorithms, also in those selected for this didactic simulator. However, for understanding in detail an algorithm, executing it step by step is the best approach but this is almost incompatible with the simulation of timers. In the present version of the simulator, timers are ignored, which has no didactic consequences for DV simulations, only minor consequences for AODV and RPL simulations, but serious consequences when one wants to compare the performance of the different algorithms, as the protocol overhead is determined by the timer settings. After lengthy discussions, the authors concluded that performance comparisons were beyond the scope of this simple simulator. Nevertheless, subsequent versions of this simulator tool will include buttons to manually simulate expiration of some timers.

2.6 Simulation algorithms

As in the Java code of the simulator, the different nodes are represented by independent objects, sending messages from one node to another simply consists in calling a method in one node from another node.

2.6.1 Distance Vector Routing

Each node has two routing attributes: (i) the optimal outgoing link to reach the sink and (ii) the cost associated with that route. Initially the cost attribute has an infinite value. Periodically, each node asks all its neighbours (nodes reachable through a single active link) the value of their cost attribute and uses the answers to update its own routing attributes. Bellmann and Ford have shown that this algorithm converges towards an optimal routing scheme [10,11]. When the network configuration changes because of node failures or changing cost of links, the answers nodes get from their neighbours may have changed and the routing scheme can start evolving towards a new optimum.

2.6.2 Ad hoc On Demand distance Vector routing

Each node has three routing attributes: (i) the optimal outgoing link to reach the gateway, (ii) the cost associated with that route and (iii) the list of incoming links that carry routes from other nodes to the gateway. No periodic updating of routing data is being done, instead, whenever a node that has no suitable routing information needs to send a message, it initiates a recursive backtracking search to find the best route to the gateway or to another node that has suitable routing information. When a node along a route fails (or when a route has not been used for some time, but this is not part of this simulator – see subsection 2.5) a recursive method resets to infinity the cost attribute of all upstream nodes. During the recursive search, whenever a route is found it can be used to transmit data. If the search discovers subsequently better routes, the original one is replaced by them.

In the present version of the simulator, the search is done by a depth first recursive backtracking algorithm. Despite the additional complexity, this will be replaced by a breadth-first algorithm, because the latter correspond better to how it is done in a real network and because it allows showing the search steps on the graph.

2.6.3 Routing Protocol for Low power and lossy network

Starting from the gateway, a tree-like routing structure is built in the network, resulting in a Destination-Oriented Directed Acyclic Graph (DODAG). A Directed Acyclic Graph (DAG) has

the property that all edges are oriented in such a way that no cycles exist. A DODAG is a DAG where all edges are contained in paths oriented toward the gateway. The building of the DODAG is initiated at the gateway by sending DODAG Information Objects (DIOs) to all directly reachable nodes. A DIO contains optional information to configure the DODAG for a specific application but contains at least a unique DODAG identifier, generated by the gateway, the cost to reach the gateway and the rank of the node that sends the DIO. This rank is initially infinite for all nodes except the gateway, whose rank is 0. When a node receives a DIO, it compares the rank in the DIO with its own rank, rejects all DIOs with higher rank numbers, because they can cause loops, sets its own rank one unit above the one of the received DIO, records the sender of the DIO as its parent and, after some delay, sends new DIOs to all nodes within reach. This process continues till all nodes are included in the DODAG. When a node has several parents, it chooses a preferred parent by comparing the cost of reaching the gateway via the different parents. When all nodes have been included in the DODAG every node can optimally send data to the gateway, just by forwarding it to its preferred parent. The preferred parent selection mechanism, based on the cost to reach the gateway, can result in a node with the same rank as its parent. This is not allowed and is avoided by increasing the rank of that node during the selection of the preferred parent. By keeping the list of parents in each node, local repairs to the DODAG can be made in case of node failure. When a local repair is insufficient to rebuild a DODAG connecting all nodes, a new DODAG building is initiated. This is triggered by a timer, but, as no timers are included in the present version of the simulator, no global repair triggering can be simulated.

2.7 Concurrency

Routing protocols operate by exchanging concurrently messages between nodes via links. As no synchronisation between nodes exists, it is quite obvious that transient states of the routing scheme will depend on minute differences in transmission and processing times. Simulating this behaviour could be done by programming the behaviour of each sensor node in a separate thread but then the transient behaviour of the simulated routing protocols would depend on the thread scheduling of the run time environment rather than on protocol properties (concurrency would be handled in the simulation time rather than in the simulated time). From a didactic point of view, an almost random transient behaviour which is not related to the protocol properties, is certainly not desirable. A full-fledged discrete event simulator, such as OMNET++, could handle these difficulties, but, as mentioned earlier, would require an excessive preliminary learning effort by the user. In the didactic simulator, sending and receiving messages is handled sequentially, node after node. In the original version of the simulator, this was done in the order the nodes were initially introduced in the graph representing the network. This fixed order sometimes resulted in reproducible (but confusing) transient behaviour of the simulated protocols. To resolve this issue, an optional randomisation of the nodes' processing order was introduced. It allows users to assert whether the observed artefacts result from the sequential processing or are from a more significant nature.

3 IMPLEMENTATION OF THE DIDACTIC SIMULATOR.

An open source package called "GrapheMaker" [14] featuring an attractive graphical User Interface was adopted as backbone for the didactic simulator. This package, written in Java, provides data structures for representing graphs and assigning attributes to the nodes and the links. Also, it implements interactive graphical methods for building the graph and setting the values of the attributes. In addition it provides methods to save graphs on file, to restore these graphs and even to create JPEG pictures of the graph. In fig.3 the graphical user interface is presented. On the left side, one can see a graphical representation of the network under study with its nodes and communication links. The top node, surrounded by a double line represents the gateway. The inside of the other nodes is partially shaded: the shading represents the state of the battery. One can notice that the batteries of nodes C and

E are much more exhausted than those of the other nodes. Links connect nodes and have, in their middle their “cost” attribute shown. The colour and the thickness of the links is used to show their role in the routing scheme. Typically coloured thick links show the best route from a given node to the gateway. To define a new network in the simulator, the buttons in the centre top of the screen are used: they allow creating and deleting nodes and links, to initialise their attributes, and to add textual comments to the graph. Dragging them with the mouse can modify the position on the screen of the different elements of the graph. The attributes can be changed at any time by double clicking the node or the link. At the right side, a table presents the actual value of the different attributes of each node shown. The contents of this table evolve dynamically during the simulation, so that the user can observe successive values of a specific attribute in a specific node, or try to understand globally the evolution of the routing state between steps of the algorithm. A running simulation can be stopped and restarted by means of the two buttons in the centre bottom of the screen. At the top, a menu bar has roll down menus to store and reload graphs (file) and to run step by step or continuously the DV, AODV and RPL collect algorithms (routing).

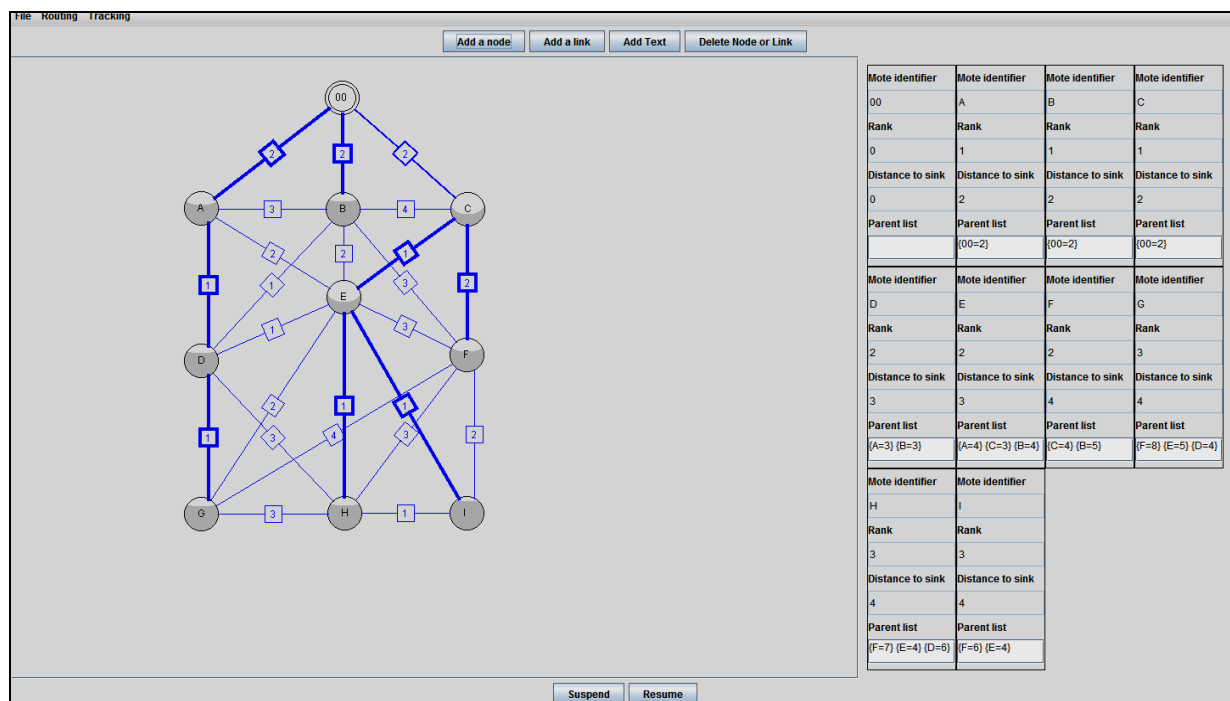


Figure 3. The Graphical User Interface of the simulator.

A third roll down menu (tracking) allows to use the same GUI to display actual routes observed in a real WSN connected via a gateway, but this function falls outside the scope of this paper.

4 EVALUATION BY STUDENTS.

This year, the subject of the BEST (Board of European Students of Technology) [15] Brussels Summer Course was “From Wireless Sensor Networks to The Internet of Things”. It gave the authors a great opportunity to have the simulator tested and evaluated by 22 engineering students who participated to this program, during a lab course in routing protocols. The students got first a theoretical introduction to the DV, AODV and RPL collect protocols and thereafter they had the opportunity to test and improve their understanding by means of the proposed simulator.

Each student was asked at the end of the session to evaluate the simulator in a quantitative way, i.e., to describe if the tool helped them to efficiently learn routing basics. First, it

appeared that a majority of students estimated that they learned more with the theoretical lesson than with their own experimentation. This might be a consequence of a technical problem: the version of the simulator installed on the computers used by the students had a poor rendering of colours while the PowerPoint slides used for the theory had well contrasted colours. Second, the evaluation provided the authors with interesting suggestions for improving the simulator. Most criticism was about the GUI: more detailed information about the state of the nodes should be visually available during simulation. This led the authors to add to the original graph the tabular representation of the value of the node attributes. Displaying more attributes in an intuitively attractive way (as it was done for the state of the battery) is certainly an extension the authors will implement in future versions of their simulator. Pacing of the simulation was also a topic that generated interesting suggestions: step-by-step and continuously running simulations, with adjustable rhythm, was perceived as extremely useful. Also, offering the opportunity to go back in time is a technically challenging but probably didactically attractive feature. Indeed, some comments stated that, when analysing the routing state after stopping the simulation, it is not obvious to understand how that state had been reached and therefore, being able to look back in time will prove helpful.

5 CONCLUSIONS AND FUTURE WORK

A simple to use simulator for learning the basics of routing protocols commonly used in wireless Sensor networks has been designed and implemented. It has been used with success in several courses, both in European and in African Universities in order to explain the “count-to-infinity” problem that affects many routing protocols. However, to use it as a tool for self-study, the first evaluations by actual students seem to indicate that additional efforts towards user friendliness and more flexibility in the simulations are required. If further evaluations by other groups of students confirm these requirements, the simulator will certainly evolve in this desired direction while focusing on presenting a high-level of simplicity and usability.

6 ACKNOWLEDGMENTS

The authors would like to express their gratitude to the Belgian Technical Cooperation (BTC/CTB) and the Fonds Jacques LEWIN - Inès HENRIQUES DE CASTRO of the Université Libre de Bruxelles for their financial support.

REFERENCES

- [1] Kay Römer, Friedemann Mattern: The design space of wireless sensor networks, IEEE Wireless Communications 11 (6):54-61 (December 2004).
- [2] The Sensor Network Museum.
<http://www.snm.ethz.ch/Main/HomePage> (Last accessed on September 16, 2011).
- [3] Mini hardware Survey.
http://www.cse.unsw.edu.au/~sensar/hardware/hardware_survey.html (Last accessed on September 16, 2011).
- [4] Vasseur, J.P. and Dunkels, A.
Interconnecting Smart Objects with IP: The Next Internet.
Morgan Kaufmann, 2010.
ISBN:978-0123751652

- [5] IEEE 802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANS), available at <http://standards.ieee.org/getieee802/download/802.15.4-2003.pdf>
- [6] JeongGil Ko, Andreas Terzis, Stephen Dawson-Haggerty, David E. Culler, Jonathan W. Hui, Philip Levis: Connecting Low-Power and Lossy Networks to the Internet, IEEE Communications Magazine 49(4): 96-101 (April 2011).
- [7] Philip Levis, Eric Brewer, David Culler, David Gay, Sam Madden, Neil Patel, Joe Polastre, Scott Shenker, Robert Szewczyk, Alec Woo: The Emergence of a Networking Primitive in Wireless Sensor Networks, Communications of the ACM, 51(7) (July 2008).
- [8] Frederik Österlind, Adam Dunkels, Jioakim Eriksson, Niclas Finne, Thiemo Voigt Cross-Level Sensor Network Simulation with COOJA. Obtained from <http://www.sics.se/~fros/osterlind06crosslevel.pdf> (last accessed on September 21, 2011)
- [9] OMNET++ version 4.0 user manual. Obtained from <http://www.omnetpp.org/doc/manual/usman.html> (last accessed on June 14, 2010)
- [10] R.E. Bellman: Dynamic programming, Princeton University Press, (1957).
- [11] L.R. Ford, Jr., D.R. Fulkerson: Flows in networks, Princeton University Press (1962)
- [12] Charles E. Perkins; Elisabeth M. Royer: Ad-hoc On-Demand Distance Vector Routing. Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, pp. 90-100 (February 1999).
- [13] JP Vasseur, Navneet Agarwal, Jonathan Hui, Zach Shelby, Paul Bertrand, Cedric Chauvenet: RPL: The IP routing protocol designed for low power and lossy networks, <http://ipso-alliance.org/white-papers> (last accessed on September 22, 2011)
- [14] Pignede Pierre-Emmanuel: Graphe Maker. Obtained from <http://licence.kaillou.net/licence> (last accessed on February 4, 2010).
- [15] BEST, Board of European Students of Technology, <http://www.best.eu.org/student/courses/event.jsp?activity=8gcl587> (last accessed on September 23, 2011).